Cooperative Task Execution between Modular Robots Based on Tight-Loose Cooperation Strategies

José Baca and Claudio Rossi and Manuel Ferre and Rafael Aracil
Centre for Automation and Robotics
Universidad Politécnica de Madrid
j.baca@etsii.upm.es, claudio.rossi@upm.es, m.ferre@upm.es

***Abstract*— The complexity in the execution of cooperative tasks is high due to the fact that a robot team requires movement coordination at the beginning of the mission and continuous coordination during the execution of the task. A variety of techniques have been proposed to give a solution to this problem assuming standard mobile robots. This work focuses on presenting the execution of a cooperative task by a modular robot team. The complexity of the task execution increases due to the fact that each robot is composed of modules which have to be coordinated in a proper way to successfully work. A combined tight and loose cooperation strategy is presented and a bar-pushing example is used as a cooperative task to show the performance of this type of system.**

I. INTRODUCTION

Nowadays, multirobot systems are an active field of research. A variety of techniques have been proposed in order to approach the problem of motion coordination in different kinds of applications [12]. Cooperation applications can be roughly divided in two classes: tight cooperation requires a continuous coordination between the robots; loose cooperation requires coordination at the beginning of the mission for planning a division of a task and at given moments of time, when re-planning may be needed. Behaviour-based [11] and schemas [1] are examples of techniques suitable for the first class of coordination problems, while market-based [5] and auction [6] techniques are commonly used in the second class of problems. The multirobots research community has focused primarily on implementing cooperative control strategies in highly efficient robots that were designed for a specific task such as wheel-based robots. However, cooperation applications with modular robot systems have been less explored. Over the last decade, research in this field has been demonstrating the potential advantages that this type of system offers such as versatility, fault-tolerance and cost-effective platforms. However, there are few studies about building multirobot teams made of modular robot configurations and evaluating the system during the execution of cooperative tasks [7] [15].

Modular robot systems are capable of forming different robot configurations made up of n-modules, which have to work in a coordinated fashion to show uniform behavior. Their ability to rearrange their modules and to adapt to different circumstances, allows them to cope with multiple tasks such as different types of locomotion and manipulation, a feature ensuring that their performance level excels in a variety of circumstances. When a robot is built by the union of several robots, such as modular robot systems, it is critical to have a tight cooperation to complete coordination of each robot configuration [17] [9], but also a loose cooperation to coordinate the colony during cooperative tasks [8] [10].

This work focuses on presenting the execution of a cooperative task by a multirobot team made of modular robot configurations. A bar-pushing example is used as a cooperative task to demonstrate that modular robots can successfully perform these types of tasks as standard robots can. A tight cooperation strategy is implemented to coordinate modules of each robot configuration and a loose cooperation strategy is used for task partition and assignment (Fig. 1). This paper is structured as follows: Section II briefly describes the modular robot system used in the experiments, named SMART. Section III points out a tight cooperation strategy implemented to coordinate modules of each robot configuration. Section IV describes a loose cooperation strategy used to divide a mission and assign to each robot its corresponding sub-task. The implementation of both strategies is detailed in section V. Section VI describes experimental results performed by modular robot configurations during an individual and cooperative task. Finally, concluding remarks are presented in section VII.

II. SMART DESCRIPTION

The SMART system [2] is a reconfigurable heterogeneous modular robot system composed of a set of interchangeable modules that form different robot or system configurations. Each module type aims to balance versatility, low-cost fabri-

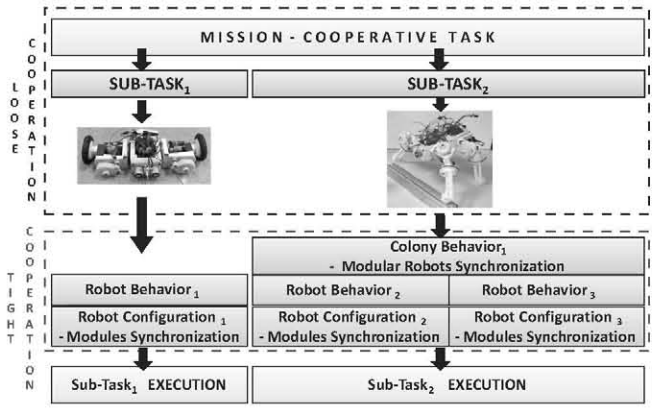


Fig. 1. Tight-loose cooperation scheme for modular robot systems.

cation and functionality. Their design permits rapid and cost effective design and fabrication. The system can be used in different applications during teleoperation tasks. The system architecture is divided into modules, M-Robots and colonies. The modules are base system components (P/C, J and S module types). An M-Robot is an autonomous entity made up of at least one P/C module and one or more J and/or S module types. Let a colony be defined as various M-Robots cooperating to fulfill a task.

The system has two communication types, i.e., Intra-robot and Inter-robot communications. Intra-robot communication refers to communication between P/C modules. Since elements are mechanically linked, it is performed via CAN bus technology. Inter-robot communication is used when no physical contact exist between P/C modules and the communication has to be carried out by Bluetooth technology.

III. TIGHT COOPERATION STRATEGY

A cooperative control strategy can be termed tight when the cooperation requires a continuous coordination between the robots. When talking about modular robot systems, the control complexity increases due to the fact that each robot configuration is built of two or more modules. Therefore, the modules have to work in perfect synchrony to behave as a single robot. The movement coordination can not only exist at robot level but also at colony level. If the cooperative task requires movement coordination of the colony, then synchronization among the M-Robots is required.

A. Synchronizing M-Robot modules

The M-Robot demonstrates robot behavior thanks to module synchronization. This global behavior is required to move the M-Robot as a unit. For example, in order to complete an action such as displacement of the M-Robot, it requires simultaneous movements across numerous modules. In other words, it would require starting and finishing all joint module movements at the same instant of time. With this type of synchronization, it is possible to create behaviors based on the robot configuration, as shown in Fig. 2. One of the problems in distributed computational systems is the lack of a global clock [4]. This is a handicap for carrying out concurrent actions in a coordinated manner. Several approaches have been proposed to overcome this problem and the one used most is the message-passing method [16]. From the computational point of view, modular robots are a set of processing units joined by a communication bus. Therefore, message-passing methods are seen as a possible solution to synchronization problems in modular robots.

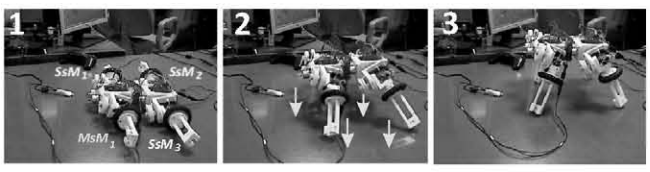


Fig. 2. Intra-robot communication allows movement coordination within the H4M-Robot modules.

A closed-loop discrete time method [3] is implemented and consists of keeping all system clocks in the same phase and period, using a single short message in every cycle. The period of that cycle can be relatively longer (seconds) than the control cycle period (milliseconds, typically the period of timer interruption). The method needs a periodic signal acting as a trigger for the closed loop. This signal is generated by the master module for every N control cycles and consists of a high priority short CAN message. In theory, all modules receive the message at the same time, but this is not correct. Each time the message is received, a local timer (tick counter) is reset and its previous values are used to correct the local timer period. When a synchronizing message enters the process, the current counter value is used to recalculate the local timer period. Consequently the counter is reset.

B. Synchronizing a colony of M-Robots

The main idea of synchronizing a colony of modular robots is the creation of a colony behavior for execution of tasks such as simultaneously pushing or lifting an object with two or more robots. Assuming that each M-Robot achieves its correct robot behavior as explained in III-A, the next step is to synchronize master modules of each M-Robot of the colony. In order to achieve the desired synchronization, a control station (PC) is used as the main task coordinator. While the master module synchronizes the slave modules, it periodically sends a signal (clk a) to the control station via BT indicating its timer value. In order to synchronize a second M-Robot from the colony, it is required to acquire a second signal from the second master module. The second M-Robot periodically sends a signal (clk b) to the computer. At this time, the control station has two signals originating from each M-Robot. The control station calculates the offset between clk b with respect to clk a, represented by equation Offset= clk $a(t)$- clk $b(t)$. Once the difference is calculated, the control station sends the correction to the corresponding master module. The master module receiving the modification adjusts its internal timer to the reference robot's internal timer. Thus, synchronization is achieved for both robots. The motion control loops are executed at the same time, thereby enabling the execution of simultaneous movements in the robots as shown in Fig. 3. The timing calculations do account for communication delays which are expected with BT message transmission. The operator can set the period of transmission of the sync signal for the robots as well as the minimum allowable gap in synchronization signals [3].

IV. LOOSE COOPERATION STRATEGY

In loose cooperation problems, a given task has to be partitioned in sub-tasks, and sub-tasks have to be assigned to individual team-members for execution. The main difference between tight cooperation problems and loose cooperation problems is that in a loose cooperation, the action of a robot has an immediate impact on the global task to be executed which affects the team-mates' goal: when a robot pushes the

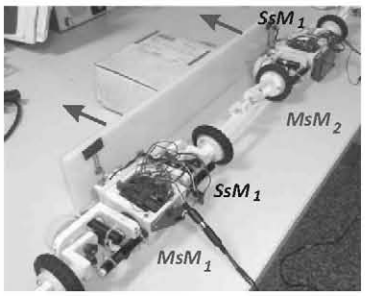


Fig. 3. Synchronizing a modular robot colony for parallel movement.

bar, the new bar position shall modify the way the other robot will push it. Our coordination algorithm is based on Rubinstein's alternate-offers protocol [14] and performs a simultaneous task subdivision and allocation in a distributed way, taking into account robot characteristics at all times.

A. Task subdivision

The task objective is to push a bar to a desired location, or equivalently, to push the bar in a desired direction. This is an instance of a more general task partitioning problem that we have generalized as follows [13]. Given a global task T_0 and r robots, the problem is how to partition T_0 into r non-overlapping sub-tasks, such as

$$T_i \cap T_j = \emptyset, \qquad \forall i,j = 1 \dots r \qquad and \qquad \bigcup_{i=1}^{r} T_i = T$$

and how to assign sub-tasks to the robots for execution. Our technique performs the two steps simultaneously and in a distributed way: each robot is aware of its own characteristics but does not know anything about its team-mates' characteristics. In our approach, the original task T_0, in this case the desired direction, is divided into two sub-tasks through a negotiation process, where each robot claims a desired push location and force such that the combination of the push actions of the two robots results in the desired direction.

A task T_i is described by a set of k parameters P_i. In this case, $k = 2$, and $P_i = \{L_i, F_i\}, i = 1, 2$. This means pushing the bar at location L_i with force F_i for Δt time[1]. Operators $\cap, \cup$ are to be defined according to the meaning of the task. In this case, considering the task parameters P_i as vectors, and taking into account standard vector addition, the union of two tasks is the projection of the sum into the desired direction vector ("useful" component), and the intersection as the projection of the addition on the vector orthogonal to the desired direction ("wasted" component), as shown in figure 4 (a).

B. Task assignment

The task negotiation process is a bargaining loop where, in turns, each robot proposes its desired push location and force, and the other computes the best push location and force for itself, taking into account the combination of the two actions, the current bar orientation, and considering that a good combination is such that the resulting direction is the one defined in T_0.

We assume that robots are willing to perform as much as they can of the given task, the only limitations being their available resources (endurance, computation power, battery consumption etc.). Thus, in a negotiation, each agent will try to maximize its reward by (i) trying to obtain a possible sub-task as large as possible (the closer to T_0 the better) and (ii) minimizing overlap with other agents' tasks (intersection, i.e. wasted effort). Each agent proposes the biggest possible share for itself, and reduces it until the counterpart finds it acceptable. Figure 4 (b) depicts the negotiation process.

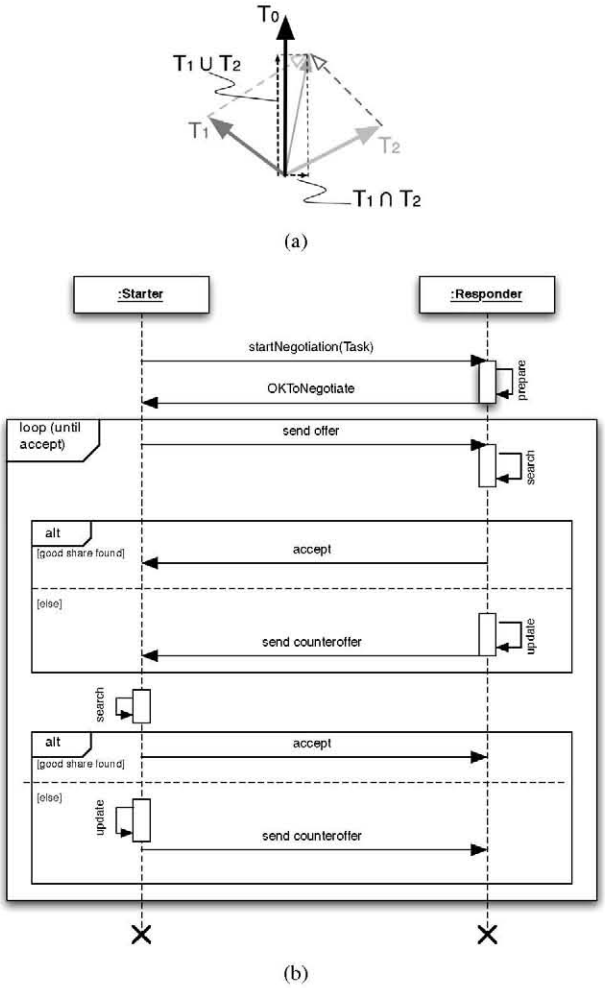


Fig. 4. (a) Union and intersection of bar pushing tasks. (b) Negotiation loop.

At the end of each bargaining loop, the robots perform the agreed pushing action. Note that since the actions eventually cause a change in the orientation of the bar, the negotiation is performed again at each outer control loop, taking into account the actual bar position.

[1]To be precise, the application of a force at a given distance from the center of mass of the bar produces a torque. Such torque, applied for Δt time produces a change of direction of the bar.

V. IMPLEMENTATION

The implementation consists of creating a control loop between the cooperative task and the modular robot colony. The main condition of this testbed is to have an autonomous team of modular robots performing a task after assigning the initial conditions. Task partition and allocation will be given by a loose cooperation strategy, and module coordination by a tight cooperation strategy.

A. Bar-pushing set-up

In a cooperative task, the main goal and the quantity of robots to use are settled in advance. In this case, a bar-pushing task has to be performed with two wheel-based robots. The scenario is composed of a bar and two W2M-Robots. Due to the fact that the robots do not contain any sensors, a video camera is placed on top of the scenario to capture the data required for the negotiation algorithm, as shown in Fig. 5.

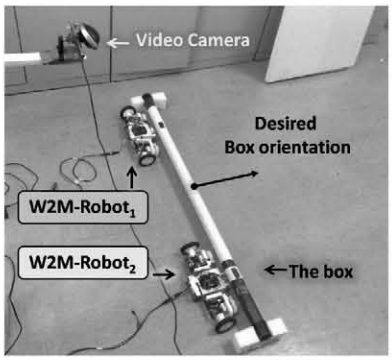


Fig. 5. Bar-pushing scenario.

B. Visual tracking interface

In order to introduce the data into a loose cooperation strategy a visual tracking interface (VTI) based on video feedback was developed. At the beginning of the experiment, the bar and each M-Robot position and orientation is manually captured by the operator by clicking over each object on the VTI (Fig. 6). The desired bar orientation is also introduced into the algorithm and can be dynamically changed during the task execution.

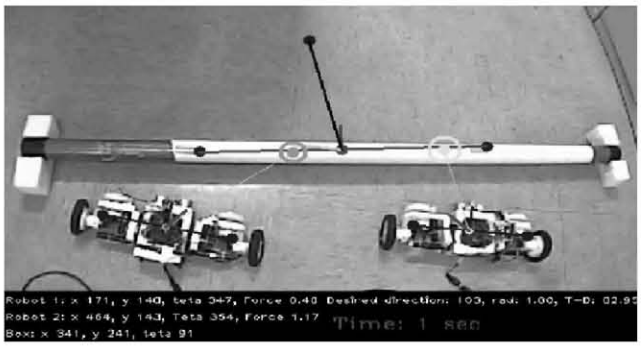
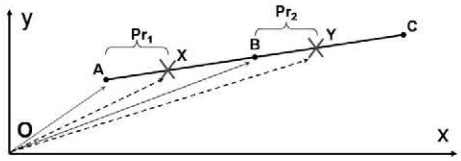


Fig. 6. Visual tracking interface.

As mentioned in section IV, the strategy divides the global task and assigns subtasks to each W2M-Robot. With these values, and the following equations (in this case for $robot_1$), the interface displays on the screen the contact point over the longitudinal axis of the bar where each robot should apply the push force. (shown in Fig.7).



$$\vec{OX} = \vec{OA} + \vec{AX}$$

$$\vec{AX} = P_{r_1} * (\hat{AB}_x, \hat{AB}_y)$$

$$\hat{AB} = (\frac{\vec{AB}_x}{\|AB\|}, \frac{\vec{AB}_y}{\|AB\|})$$

$$\|AB\| = \sqrt{\vec{AB}_x^2 + \vec{AB}_y^2}$$

Where:

$\vec{OX}$ = Push force contact for $robot_1$

P_{r_1} = Value generated by the negotiation algortihm.

Fig. 7. The x and y represent the position where robots apply the force.

C. Modular robot graphical user interface

The autonomy of the system is complete when the negotiation algorithm is linked with the VTI and the modular robot graphical user interface (MR-GUI). Normally, the MR-GUI is in charge of teleoperating and assigning behaviors to modular robot configurations. After linking the MR-GUI with the VTI, no operator is required for the cooperative task execution. The MR-GUI automatically controls and assigns the corresponding operating modes to the W2M-robots. The correct execution of the operating mode will depend on the tight cooperation strategy explained in section III.

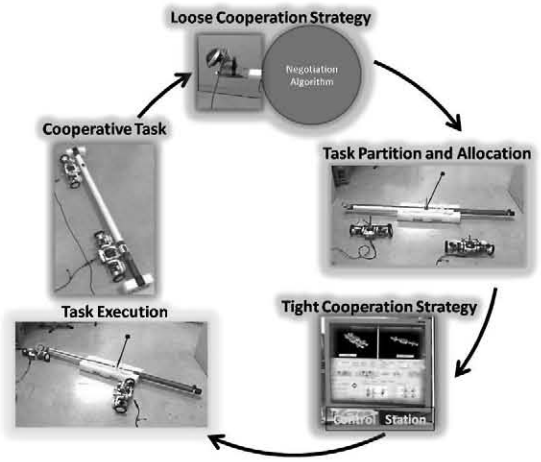


Fig. 8. Cooperative task execution is performed according the negotiation algorithm demands. Once the subtasks have been achieved, the new position and orientation parameters from the objects are uploaded to the negotiation algorithm and the cycle begins once again until task fulfillment.

VI. EXPERIMENTAL RESULTS

This section describes the experiments that show the modular robots' performance during the execution of a cooperative task. A bar-pushing task is used as an example to explore the cooperation between two wheel-based modular robot configurations.

A. Tight cooperation experiment

The first experiment attempts to demonstrate that the W2M-Robot configuration can satisfactorily perform a behavior that a standard robot of its kind can do. The robot configuration used in this experiment is composed by one P/C module, two J modules, two S modules, and two software modules (MsM and SsM). Each software module loaded on each side of the P/C module controls its corresponding J module, therefore a tight cooperation between the master and the slave modules has to be carried out. The assembled W2M-Robot configuration should behave as a differential drive mobile robot, that is, forward, backward and rotation movements have to be performed in a proper manner to satisfy the demonstration. Using the VTI described before, the robot position and orientation parameters are captured into the tracking system. Three movement types were executed by the W2M-Robot as shown in Fig.9.

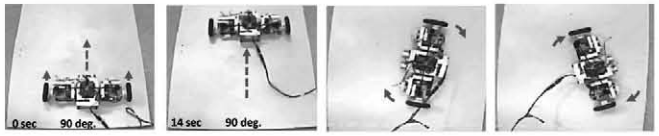


Fig. 9. The W2M-Robot configuration moves as a differential drive mobile robot. Synchronization of modules is required to achieve correct robot behavior.

The initial robot orientation is compared every second while performing the movement. For instance, in a forward displacement during 14 seconds (without considering gear backlash from the J module, and video feedback error) the robot orientation should not change over the time. The experimental results in Fig.10 show that the W2M-Robot configuration achieves a satisfactory performance with an average robot-orientation error of 1.7 degrees.

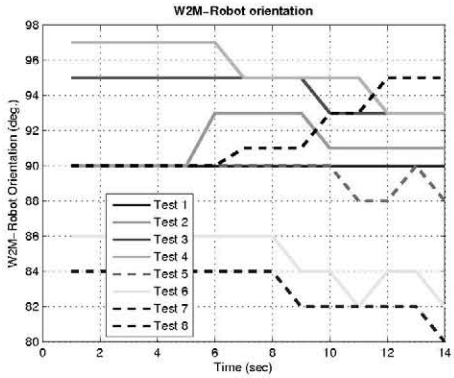


Fig. 10. Eight tests performing forward displacement during 14 seconds displays an average error of 1.7 degrees.

B. Loose cooperation experiment

The intention of the second experiment is to demonstrate that a modular robot colony can satisfactorily perform a cooperative task, just as similarly specialized robots can. In other words, the modular robot colony will be controlled by a loose cooperation strategy which was designed for specialized robots. Two types of experiments were performed for this demonstration.

1) Pushing the bar to a desired orientation: The objective is to push a bar to a desired orientation. As it can be seen in Fig.11 the negotiation algorithm divides the mission in subtasks to each robot to complete the goal. Each robot configuration performs its task in an autonomous way until the bar orientation matches the desired orientation. It should be noted that the experiment did not require both robots to push the bar at the same time. The solution given by the loose cooperation strategy is that only W2M-Robot$_2$ should apply the pushing-force for completing the mission. The tracking system displays bar and robots position path. It can be noticed that W2M-Robot$_1$ keeps close to the bar but never pushes the bar. A total of 50 seconds were required for obtaining desired bar orientation.

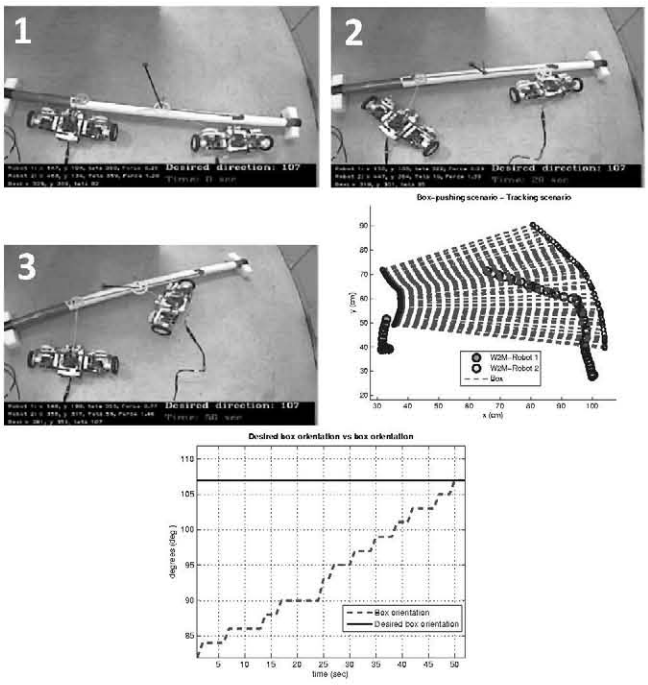


Fig. 11. The strategy decides the most efficient way for task achievement.

2) Pushing the bar with desired orientation: This experiment emphasizes the next step after obtaining the desired bar orientation. The objective is to push the bar while maintaining the desired bar orientation, as shown in Fig. 12. The complexity of the task increases due to the fact that both robots should simultaneously move in a coordinated fashion to achieve the goal. It can be seen during the execution of the task a satisfactory performance of both W2M-Robots. During 25 seconds, an error of 4 degrees between the desired bar orientation and bar orientation is produced.

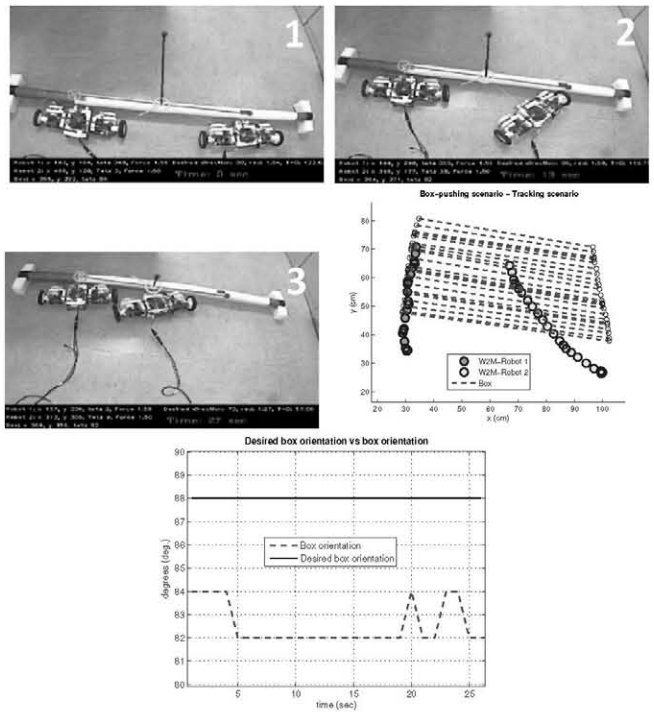


Fig. 12. High coordination and satisfactory performance from both W2M-Robots are necessary to complete the cooperative task.

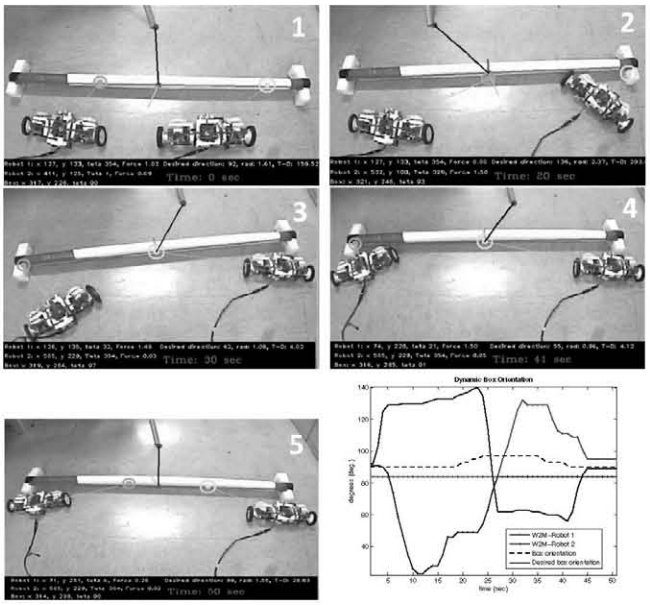


Fig. 13. The desired bar orientation can be dynamically changed as seen in this experiment. This is a real time process and the corresponding modifications begin immediately.

VII. CONCLUSIONS AND FUTURE WORK

The combination of both tight and loose cooperation strategies demonstrates that modular robot systems can successfully perform a cooperative task as standard robots can. An individual and cooperative experiments were executed to displays each strategy's performance. With a tight cooperation strategy, the M-Robot exhibits appropriate behavior with respect to its robot configuration. The modules achieve proper synchronization for the execution of coordinated movements. A bar-pushing experiment was used to demonstrate the execution of a cooperative task with a team of modular robot configurations. Our future work is to continue demonstrating that modular robots can be suitable systems for performing cooperative tasks. The implementation of new cooperative strategies and the execution of different cooperative tasks are the next goals.

VIII. ACKNOWLEDGMENTS

The authors would like to thank CICYT and CONACYT for financing this research project. Also special thanks to Erik Hernandez and Cecilia Garcia for their contribution during the execution of experiments.

REFERENCES

[1] R. C. Arkin. Cooperation without communication: Multiagent schema-based robot navigation. *Journal of Robotic Systems*, 1992.

[2] J. Baca, M. Ferre, R. Aracil, and A. Campos. A modular robot system design and control motion modes for locomotion and manipulation tasks. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010.

[3] J. Baca, M. Ferre, M. Collar, J. Fernandez, and R. Aracil. Synchronizing a modular robot colony for cooperative tasks based on intra-inter robot communications. In *Proc. IEEE Int. Conf. on Electronics, Robotics and Automotive Mechanics*, 2010.

[4] Randy Chow and Theodore Johnson. *Distributed Operating Systems and Algorithms*. Addison-Wesley, 1997.

[5] M Bernardine Dias and Anthony (Tony) Stentz. Traderbots: A market-based approach for resource, role, and task allocation in multirobot coordination. Technical Report CMU-RI -TR-03-19, Robotics Institute, Pittsburgh, PA, August 2003.

[6] B. P. Gerkey and M. J. Mataric. Sold!: Auction methods for multirobot coordination. *Trans. on Robotics and Automation*, 18:5, 2002.

[7] K. Matsumoto, H. Chen, J. Ota, and T. Arai. Automatic parameter identification for cooperative modular robots. In *Proc. Int. Symposium on Assembly and Task Planning*, pages 282–287, 2001.

[8] G. T. McKee and P. S. Schenker. Networked robotics. *Sensor Fusion Decentralized Control In Robotic Systems III*, 4196:197–209, 2000.

[9] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji. M-tran: self-reconfigurable modular robotic system. *IEEE/ASME Transactions on Mechatronics*, 7(4):431–441, 2002.

[10] L.E. Navarro-Serment, R. Grabowski, C. J. J. Paredis, and P.K. Khosla. Millibots. *IEEE Robotics and Automation Magazine*, 9(4):31–40, December 2002.

[11] L.E. Parker. L-alliance: Task-oriented multi-robot learning in behaviour-based systems. *Advanced Robotics*, Special Issue on Selected Papers from IROS'96, 1997.

[12] L.E. Parker. Current research in multi-robot systems. *J. Art. Life and Robotics*, 7, 2003.

[13] C. Rossi, L. Aldama, and A. Barrientos. Simultaneous task subdivision and allocation for teams of heterogeneous robots. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 946 –951, 12-17 2009.

[14] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50:1, 1983.

[15] Liying Su, Lei Shi, and Yueqing Yu. Collaborative assembly operation between two modular robots based on the optical position feedback. *Journal of Robotics*, 2009.

[16] J. Xu and K. Hwang. Heuristic methods for dynamic load balancing in a message-passing supercomputer. In *Proceedings of Supercomputing*, pages 888–897, November 1990.

[17] Mark Yim, Ying Zhang, K. Roufas, D. Duff, and C. Eldershaw. Connecting and disconnecting for chain self-reconfiguration with polybot. *IEEE/ASME Transactions on Mechatronics*, 7(4):442–451, December 2002.